datamaster.library General documentation

COLLABORATORS			
	TITLE : datamaster.library General documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 24, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

datamaster.library General documentation

1.1 datamaster.library General documentation

datamaster.library General documentation

datamaster.library v2.0

$VER: May 1997

1.2 Table Of Contents

1.3 Introduction

1. Introduction

datamaster.library is a standard Amiga Shared library.
None of the existing recognition libraries suited my needs, so I decided
to create mine (good old programmer's behaviour ;-)
So...What is "so special" in datamaster?
Well, here are its main features:
* A MODULAR architecture,with external recognizers, that can be very
easily listed, added, removed,etc... by user, with the DMcontrol
program.
These recognizers are real subprograms, not old classic user-defined
strings!
This means two things:
---
1) Such recognizers are extremely FLEXIBLE, being made of "real
code", they can recognize ANY filetype.
As an example of the recognizer's flexibility,there is even a
LINK-Datatypes recognizer,that uses the datatypes.library!
It's just in case you find brand new datatypes you REALLY need,

and there is no existing recognizer (yet ;-),so you can use this
'bridge' to datatypes meanwhile.
2) YES, YOU users won't be able to spend nights defining
"Matchbyte=XXX" filetypes anymore :)
BUT....don't worry, as:
- The current "bank" of recognizers surely suits 99% of
your needs :)
- Programming a recognizer is rather trivial.
- If there is feedback (I hope so :), there will be updates
of recognizer files, exactly like for external music
players, or xpk sublibraries.
* Small & fast library (100% assembler).
(The library itself is 6Kb or so!!...then you add/remove the recognizers
YOU need).
* The recognizers are sorted by priority:recognizers that are known to be
slow/rare, or handling very much filetype will have a low priority
etc etc... So----->speed gain.
* datamaster.library offers 2 filetype checking modes:
->REAL DATA checking ("heavy" mode,but very precise).
Whatis.library does it too,it's true,but datamaster uses external
recognizers using >>REAL CODE<<.This should beat all the
MATCHBYTE(0x44ff00..) thing I guess....And recognizers come out
with a 100-200 bytes size which is not that much you'll agree!
->FILE PATTERN matching,very classic,used in most the previous
libraries of this kind before.(accuracy:so,so :-)
For programmers:
~~~~~~~~~~~~~~~~
* The main recognition functions are very easy to use.
(ONE library call is enough :)
* Many support functions.
These functions are for all kind of text/string comparisons,search etc...
They were made to be used in the recognizers, but they may be used as
well in your own programs.
Note: being an assembly programmer, I thought of these functions for asm
users. (C users already have most of these in amiga.lib)

1.4 Requirements/Installation

2. Requirements/Installation
Requirements
Installation
~~~~~~~~~~~~~~~~

1.5 Requirements

2.1. Requirements

This library needs:

o AmigaOS 2.04 / Kickstart V37 or higher.

o Reqtools.library (optional) ->error messages.

(© Nico Francois)

o xpkmaster.library (highly recommanded) ->datamaster.library/Testfile()

(© Bryan Ford, Urban Dominik Müller, Christian von Roques, Dirk Stöcker)

1.6 Installation

2.2. Installation

Very simple...Just copy all the files of the 'libs' directory in LIBS: .

- LIBS:

- Recognizers (Dir)

- Store (Subdir) <-where to put all non-used recognizers.

- all recognizer files

- datamaster.library

User has a direct control on the library with the DMcontrol program.

1.7 Important concepts.

3. Important concepts.

Here are some global informations about the different parts of this librarie's package,how it works,etc...

Docs to create the recognizers,& for using the library functions should be found somewhere else in this package.

But here you'll find things that aren't explained elsewhere so please read it all ;-)

DATA checking & PATTERN matching.

Recognizers & Filetypes

Library's initialization

1.8 DATA checking & PATTERN matching.

3.1. DATA checking & PATTERN matching.

As already said,this library provides 2 checking-modes.

DATA Checking

PATTERN matching

1.9 DATA Checking

3.1.1. DATA Checking

This is the most powerful mode.This is why this library was created in
fact,as I didn't like datatypes (heavy,slow... ;-) & really needed
data-check.
For programs that ALREADY HAVE LOADED SOME DATA & then,want to know what
type it is,this is the >ONLY< solution.
The recognizers that support data checking include some code that does
the job;the library just passes the data to the check-routine of each of
all these recognizers,one by one....

1.10 PATTERN matching

3.1.2. PATTERN matching

More 'classic' mode.This is how most of the existing recognizing tools
work (automatic recognitions of DirOpus,DirWork,other libraries etc...).
It just compares the filename with some pattern,specific of the
filetype.This is not always 100% reliable,for example if the file doesn't
have THE specific extension,or even not particular sign at all! But it's
handy & often quick so I've added it anyway.
Programs dealing with >>FILE<< recognition may rely on this only,or on
data checking by loading little bits of the file (the library provides
direct functions for this ;-).It is very important that programmers who do
such FILE-recognition give the user the CHOICE of the checking-mode.More
explanations in Recognizers & Filetypes

1.11 Recognizers & Filetypes

3.2. Recognizers & Filetypes

It's important to define precisely these :

Filetype

Recognizer

1.12 Filetype

3.2.1. Filetype

When the library analyzes data/filename,it will give you back a filetype;
If data doesn't match any of the available filetypes,then you will get

the ’Generic’ filetype.(<=> unknown data)

There are MAJOR-FILETYPES and SUB-FILETYPES. (VERY simple, don’t panic! :)

MAJOR-FILETYPES:

~~~~~~~~~~~~~~~~

When you do a "DMcontrol LIST", you obtain the list of "MAJOR" FILETYPES.

By convention, they have the same name as the recognizer files.

Example: there is a "PACKED-LHA" recognizer, then the "PACKED-LHA" filetype

exists. No problem eh?

But some recognizers are more complex and can give you more precise

informations on the file type...

SUB-FILETYPES:

~~~~~~~~~~~~~~

...So, some filetypes are "hidden" in "more global" recognizers.

Example:

The "SMPL-Others" recognizer detects ADPCM, AIFF, SUN, VOC and Wave samples.

Conclusion: when you have to define filetypes entries in some prefsfile, in

such case, you can either use "SMPL-others", or "AIFF", "SUN" etc...

separately. The advantage is obvious I guess!

Note: You don’t always have access to such sub-filetypes:

- They’re only implemented if they it detecting them separately can be

useful.

- They can only be implemented on DATA-checking modes.(NOT patterns)

"But how can I know all the filetypes datamaster recognizes?"

~~~~~~

-------> Have a look at the Usage section.

Note: filetype names are case-independent.

1.13 Recognizer

3.2.2. Recognizer

The recognizers are sub-programs which job is...to recognize filetypes.

But as explained in the Filetype section with the "SMPL-others" example,

it’s crucial to understand that ONE recognizer can detect MORE THAN JUST

ONE filetype.

As a convention, the recognizer files, in the "LIBS:Recognizers/",

directory, "give their name" to the corresponding MAJOR-FILETYPES.

The MAJOR-FILETYPE names should usually have the syntax: "CLASS-name".

(ex:PACKED-LHA, ANIM-FLI etc...)

Note: these names are stored IN the recognizers in fact.
~~~~~~

(you can try to modify completely all the recognizer filenames, it won't change anything. The names are just the same for "eye" reasons :-)
A bit tricky, now: datamaster enables you to create many recognizers with same MAJOR-filetype name. Why? Simple: you can easily create add-ons for existing MAJOR-filetypes.
To stick to our example, you could create a "MUSIC-Others.mynewstuff" recognizer FILE. (But of course, its internal filetype name would be "MUSIC-Others").
NOTES:

This way of using executable files has both advantages & drawbacks:
* Advantage:the DATA checking routines are very powerful as they are in done in real code which makes them more flexible than any pseudo-basic language used in some recognition tools,which suffice for simple position comparisons at some particular offsets,but can't deal with hard-guess formats :-)
* Advantage:As real code is used,the recognizers can also use other libraries resources (xfdmaster for instance),etc etc...
* Drawback:Some simply-recognized formats (1 or 2 offset matches for example),make the recognizer a bit 'big' for what it is,as the routine is very small but the recognizer file is still 150 or 200 bytes (Wow that's not too big eh?).Keep in mind that executable files contain relocation hunks,so the recognizer takes in fact LESS space in memory than the file on disk.
* Drawback:such compiled/assembled files are less 'user-editable' for sure.All you need is a VERY VERY basic assembler knowledge to make simple recognizers,but every user doesn't program which is perfectly understandable.Well,in that case,find a friend who does,or send me the format you'd like to be added.

1.14 Library's initialization

3.3. Library's initialization
When it opens, the library will scan the default dir LIBS:Recognizers/ and load & initialize all the recognizers it finds in it.
This may take some time. That's why there is a "CREATEBUNDLE" command in DMcontrol. This will merge all the recognizers of the directory into one "big" file. If the library finds such a file, it will use it instead of scanning all the recognizer files. This will be faster (specially if you have lots of recognizer files).
Of course don't forget to "refresh" this file each time you install new recognizers.

1.15 Usage

4. Usage

Knowing the available filetypes

LINK-Datatypes recognizer

Other stuff

1.16 Knowing the available filetypes

4.1. Knowing the available filetypes

What YOU, user want to know is just what filetypes you datamaster can
currently detect.

If you hate reading docs, here is the easy way...

use "DMcontrol TypeInfo=all"... All the output in BOLD WHITE text can be
used as filetype string :-))

Displaying the MAJOR-Filetypes:

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

"DMcontrol LIST"

You'll obtain something like:

>>28<< Filetypes available:

ANIM-FLI

ANIM-IFF

ANIM-MPEG

ASCII-AmigaGuide

ASCII-REXX

ASCII-Text

Executable

Generic

GFX-BMP

GFX-DXF

...etc...

How to get also the SUB-Filetypes (if any) :

~~~~~~

for a particular major-filetype:

"DMcontrol TypeInfo=<name>" (or "DMcontrol ti=<name>)

for the whole list:

"DMcontrol TypeInfo=all" (or "DMcontrol ti=all)

Note that TypeInfo will give some other info on the recognizers...

Click lhere for the complete documentation of the DMcontrol tool.

1.17 LINK-Datatypes recognizer

4.2. LINK-Datatypes recognizer
This recognizer is VERY special, as it relies on the datatypes system to
get sub-filetypes.
What for you may ask?
Well, it has a very low priority, so it will be used in very last case, if
all "normal" recognizers have "failed" :-)
A nice feature is that datatypes will be just used for tests, and WILL
NOT STAY IN MEMORY once it’s done: this will not cost you any extra memory.
The problem is that it’s not possible to obtain a LIST of all the
sub-filetypes the datatypes can provide :-(
The strings will only be available when some corresponding data is
recognized! :-(
So the only solution is to discover these aditional sub-filetypes when
they "appear"...

1.18 Other stuff

4.3. Other stuff
If you don’t need all recognizers, simply move the files to the
Recognizers/Store directory and that’s all!!
(before library loading of course :-)
If you want to remove recognizers/filetypes "on the fly" you will have to
use DMcontrol.
Click here to see its documentation.
"DM_PattPri" ENV_Variable:

When a program asks the library to recognize something, it can request to do
it by "real data check", "pattern check", or BOTH. In the last case, both
are tried, and if both don’t get a "generic" result, then the DATA-Check
filetype will be used.(considered as more reliable)
If you set the ENV-variable DM_PattPri to 1, this will FORCE datamaster to
prefer the "pattern-check" instead, in this situation.
(This may be useful in some cases. Everything should work fine without it,
though :-)
The variable is dynamic, you can change it as much as you want (no need to
reboot or flush the lib).

1.19 Legal mumbo Jumbo

5. Legal mumbo Jumbo

Disclaimer

Distribution

1.20 Disclaimer

5.1. Disclaimer

This program is provided "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of this program is with you. In no event can I be liable to you for damages of any kind arising out of the use of this program, or the inability to use it.

1.21 Distribution

5.2. Distribution

o datamaster.library is Copyright ©1996-1997 by Alexis Nasr. All rights are reserved.

o datamaster.library is "freeware" so no donation is required. It is *not* in the public domain.

o datamaster.library may be freely distributed provided all the files of the original package remain unaltered and are included in the distribution. They may, however, be archived to conserve space.

o No profit is to be made by selling this software. You may only charge enough to cover reasonable production and distribution costs.

o This software may not be included in a commercial package, or on a magazine coverdisk, without the author's written permission.

o This software may not be uploaded onto any BBS that claims copyright on uploaded material.

o If you use this package, I would enjoy receiving a postcard or anything you'd think worth from you. See the Author paragraph.

1.22 Some infos

6. Some infos

My config:

---

* A1200+ Blizzard A1230 IV Card (68030/50Mhz/16Mb/FPU)

* HD 540 Mb

* Squirrel + CD300e x2

* 1085S & good'old DF1: ...what a joy when I had bought it :-)

Development:

Entirely in assembler,using the hmm... <NO COMMENT> Asm-One v1.29 ;-)

It has been succesfully tested with Enforcer & Mungwall.(the program,of

course,NOT Asm-one....just joking...really?)

This guide file has been generated by Text2Guide 03.10 © Stephan Suerken.

(I had enough of making these guide files by hand!)

1.23 History

7. History

The versions refer to the WHOLE PACKAGE.

Library version/revision changes are signalled when necessary.

v2.0 Release:

~~~~~~~~~~~~~

Should be this one, hopefully :)

v2.0 Beta 3:

~~~~~~~~~~~~

* Added the new sublist concept, with DMR_SubTypeTable &

DMR_CheckRoutineHook tags.

* Added the dmTestData() function.

* Added the DM_PattPri env-variable.

* Autodocs improved (I think :)

* "MUSIC-*" recognizers highly enhanced

* All internal recognizers are "out" of the lib now. Nicer...

Executable recognizer has been enhanced too.

v2.0 Beta 2:

~~~~~~~~~~~~

* "bundle" file handling feature implemented.Added HOOKNAME tag feature.

* Autodocs improved. Docs too.
~~~~~~~~~~~~

* Includes: asm fixed, C created. (thanx Dirk ;-)

* Small Bugfixes in master library.

* recognizers:

~~~~~~~~~~~~

- NEW: * ANIM-AVI,ANIM-QT (Solo)

* PACKED-HotHelp (Dirk Stoecker)

- BUGFIXES:

* ASCII-Text:accepted too many non-really-ascii data.

* ASCII-AREXX: messed with C files & others....

* Misc stuff changed in other recognizers.

v2.0 Beta 1:

~~~~~~~~~~~~

-> datamaster.library v2.0

* Bugfixes in the library and associated progs.

* New functions, tags etc added.

* New recognizers, & "MUSIC-*" ones highly enhanced.

* Documentation "freshened" :)

v1.1: (unreleased)

~~~~~

* Bugfixes in DMlauncher.

* Bugfixes/improvements/additions in the recognizers.

v1.0:

~~~~~

Initial version.Developer info not totally complete.

1.24 Future - Creating new recognizers

8. Future - Creating new recognizers

The library itself will certainly evoluate,but it’s self-evident that

most of the evolution potential is in the recognizers.So if there is a

format you’d like to be added,just send me:

* as much doc as you have on the format itself (if you have it ;-)

* as much files as possible of this format

* explanations on what this format is used for etc...

If you do yourself a recognizer,of course,please send it! It’ll be added

to the next package release.(Please send me also the format description,

the sources, testfiles etc...) [asm prefered ;-]

1.25 Other documentations of this package

9. Other documentations of this package

The current recognizers

DMcontrol ,an essential program for using the library

DMlauncher ,a simple yet very efficient Bonus-program

1.26 Author

Postcards are welcome...thank you (never received much feedback with my other programs but I just continue asking as I’m a bit stubborn.

Thanks fly to:

* Olivier Dewee for his work on the C includes a year ago.

* Dirk Stoecker for adapting the C includes, & making the HotHelp C example.

* Henryk "Buggs" Richter for his "Playerloader" sources (you saved me "some" work, for sure, pal! ;-))

* Solo, for providing me docs on some ’weirdo’ formats (TIFF,PCX,FLI,DXF,BMP,ZIP & all the hell! :).

(tu te rends compte, t’as contribué a du code "utile" c’est fou tu trouves pas? ;-))

...And also for the bugreports & recognizer-job ;)

Now some adverts for some of my other progs:

~~~~~~

* Surveymem v3.08: (util/cdity)

* TFMX-converter v1.0: (mus/misc)

* Chipsaver v1.83: (mus/misc)

* XpkCybPrefs: (util/pack)

Well, that’s all folks! Have fun, Amiga rulez!

Snail Mail: The Cyb’

Alexis Nasr E-mail: nasr@hol.fr

27, rue Formigé ------

Résidence RENOIR French Minitel network: 3615/RTEL1

33110 Le Bouscat ---------- BAL: "The Cyborg/NGC"

(FRANCE)
~~~~~~